IN THE SPECIFICATION

Please amend the paragraphs of the specification as follows:

Please replace paragraph [00013] on page 3 with the following paragraph:

[00013] Various embodiments of the invention may be incorporated in a wireless communication system operating in accordance with the code division multiple access (CDMA) technique which has been disclosed and described in various standards published by the Telecommunication Industry Association (TIA), Third Generation Partnership Project (3GPP) and Third Generation Partnership Project 2 (3GPP2). Such standards include the TIA/EIA-95 standard, TIA/EIA-IS-856 standard, IMT-2000 standards (including cdma2000 standards and WCDMA standards), all incorporated by reference herein. A copy of the standards may be obtained ~~by accessing the world-wide web at the address: http://www.3gpp2.org, or~~ by writing to TIA, Standards and Technology Department, 2500 Wilson Boulevard, Arlington, VA 22201, United States of America. The standard generally identified as WCDMA standard, incorporated by reference herein, may be obtained by contacting 3GPP Support Office, 650 Route des Lucioles-Sophia Antipolis, Valbonne-France.

Please replace paragraph [00014] on page 4 with the following paragraph:

[00014] Generally stated, a novel and improved method and an ~~accompanying~~ accompanying apparatus provide for efficient decoding of a sequence of encoded data symbols in a communication system. One or more exemplary embodiments described herein are set forth in the context of a digital wireless data communication system. While use within this context is advantageous, different embodiments of the invention may be incorporated in different environments or configurations. In general, the various systems described herein may be formed using software-controlled processors, integrated circuits, or discrete logic. The data, instructions, commands, information, signals, symbols, and chips that may be referenced throughout the application are advantageously represented by voltages, currents, electromagnetic waves, magnetic fields or particles, optical fields or

particles, or a combination thereof. In addition, the blocks shown in each block diagram may represent hardware or method steps.

Please replace paragraph [00021] on page 7 with the following paragraph:

[00021] An algorithm commonly known as [[MAP]] Maximum Aposteriori Probability algorithm may be used in the decoding process. The decoding process in trellis 200 may include determining state metrics and path metrics from the initial state space and final state space at the same time in a forward and backward fashion. A soft decision determines the values of the encoded data symbols X_i and Y_i as represented by the branches at time instances corresponding to the state spaces in the trellis. To make a soft decision for the values of the encoded data symbols X_i and Y_i, a dual metric (DM) of each branch is determined. The DM of a branch is based on a forward state metric (FSM), a backward state metric (BSM) and a branch metric (BM). The DM may be a summation of FSM, BSM and BM. BM of a branch may be determined based the channel output.

Please replace paragraph [00030] on page 12 with the following paragraph:

[00030] Referring to FIG.6, a symbol X_i node 601 may be a symbol node associated with the data symbols at the input of first constituent code 401 at time "i." A symbol Y_i node 602 may be a symbol node associated with the data symbols at the output of first constituent code 401 at time "i." A symbol Z_k node 603 may be a symbol node associated with the data symbols at the output of second constituent code 402 at time "k". The state nodes of the trellis associated with first constituent code 401 at times "i-1" and "i" may be represented by respectively a state S_{i-1} node 604 and a state S_i node 605. The state nodes of the trellis associated with second constituent code 402 at times "k-1" and "k" may be represented respectively by a state σ_{k-1} node 606 and a state [[σk]] σ_k node 607. The computational nodes associated with the first constituent code 401 at times "i-1, "i" and "i+1" may be respectively represented by a C_{i-1} node 608, a C_i node 609 and a C_{i+1} node 610. The computational nodes associated with the second constituent code 402 at times "k-1, "k" and [["K+1"]] "k+1" may be represented respectively by a D_{k-1} node 611,

a [[Dk]] D_k node 612 and a D_{K+1} node 613. A channel R_x node 614 is associated with the received data symbols X_i. A channel R_y node 615 is associated with the received data symbols Y_i. A channel R_z node 616 is associated with the received data symbols Z_k.

Please replace paragraph [00034] on page 14 with the following paragraph:

[00034] The channel nodes [[614-16]] 614, 615 and 616 are updated by receiving a channel output associated with the corresponding data symbols. A correlator in the demodulator 19 may output the channel output for each data symbol. For example, channel outputs associated with data symbols X_i, Y_i and Z_k update respectively channel nodes 614, 615 and 616. When a channel node is updated, a message is passed from the channel node to the associated symbol node. For example, in case of data symbol X_i, after receiving an X_i channel output at channel node 614 associated with data symbol X_i, channel node 614 is updated by passing a message from the channel node 614 to symbol node 601. The message passed from channel node 614 to symbol node 601 may be equal to the log-likelihood ratio: $M(R,x_i)=\log\left(\frac{P(x_i=1|R)}{P(x_i=0|R)}\right)$ of the symbol X_i channel output. For example, if the symbol X_i is BPSK modulated such that 1 is transmitted as a positive voltage, and 0 is transmitted as a negative voltage, and received over an AWGN channel as a voltage, represented by the real number r_i, then the message may be equal to: $M(R,x_i)=\log\left(\frac{P(x_i=1|r_i)}{P(x_i=0|r_i)}\right)=4\left(\frac{E_s}{N_o}\right)\cdot r_i$, where E_s/N_o represents the symbol SNR of the channel. For a data symbol at an instance of time, the channel node need only be updated once. The channel node may be updated by receiving the channel output.

Please replace paragraph [00037] on page 17 with the following paragraph:

[00037] A state node may also be updated by receiving a message on the connected branches. To update a state node, the state node passes a message received from a connected computational node on an incoming branch to another connected computational node on an outgoing branch. The state nodes are connected only to the

computational nodes. The state nodes basically pass the incoming messages from a computational node to another computational node. The state S_i node 605 is connected to computational C_i node 609 and C_{i+1} node 610. The state S_{i-1} node 604 is connected to C_{i-1} node 608 and C_i node 609. When state S_i node 605 is updated by receiving a message $M(C_i, S_i)$ on an incoming branch 665 from C_i node 609, the message $M(C_i, S_i)$ is passed on as a message $M(S_i, C_{i+1})$ on an outgoing branch 667 to C_{i+1} node 610. When state S_i node 605 is updated by receiving a message $M(C_{i+1}, S_i)$ on an incoming branch 668, the message $M(C_{i+1}, S_i)$ is passed on as a message $M(S_i, C_i)$ on an outgoing branch 666 to C_i node 609. Similar operations are performed when updating state S_{i-1} node 604, state [[σk]] $\underline{\sigma_k}$ 607 and state σ_{k-1} node 606.

Please replace paragraph [00041] on page 20 with the following paragraph:

[00041] An outgoing message from computational node Dk 609 to symbol node Z_k may not exist even though a branch 654 (in dotted line) is shown. Since symbol node Z_k 603 sends a message to only ~~Dk node 603~~ $\underline{D_k}$ node 612, the node Z_k 603 does not receive a message from ~~D_k node 603~~ $\underline{D_k}$ node 612.

Please replace paragraph [00048] on page 21 with the following paragraph:

[00048] Referring to FIG. 7, a graph 700 depicts flow of messages between different nodes corresponding to different instances of time in accordance with various embodiments of the invention. The channel output associated with encoded data symbols X_i, Y_i and Z_k may be stored in a memory element in demodulator 19. As such, the channel output associated with encoded data symbols X_i, Y_i and Z_k for different time instances may be available all at the same time for the decoding process. The channel outputs associated with data symbols X_{i-1}, X_i and X_{i+1} corresponding to time instances "i-1", "i" and "i+1" are passed on to symbol nodes 701. The state S_{i-2}, S_{i-1}, S_i and S_{i+1} nodes 702 may also be formed to connect the computational C_{i-2}, C_{i-1}, C_i, C_{i+1}, and C_{i+2} nodes 704. The state σ_{k-2}, σ_{k-1}, [[σk]] $\underline{\sigma_k}$ and σ_{K+1} nodes 705 may also be formed to connect the computational D_{k-2}, D_{k-1}, [[Dk]] $\underline{D_k}$, D_{K+1}, and D_{k+2} nodes 706. The channel outputs associated with data

symbols Y_{i-1}, Y_i and Y_{i+1} corresponding to time instances "i-1", "i" and "i+1" are passed on to symbol nodes 707. The channel outputs associated with data symbols Z_{k-1}, Z_k and [[Z_{K+1}]] Z_{k+1} corresponding to time instances "k-1," "k" and [["K+1"]] "k+1" are passed on to symbol nodes 708. Although the graph 700 depicts the message passing flow corresponding to three instances of time, one ordinary skilled in the art may appreciate that the graph 700 may be expanded to included all the time instances for 1 through N. In accordance with an embodiment, all the computation nodes 704 and 706 may be triggered essentially concurrently. As such, in one step all the computational nodes are once updated. Each time all the computational nodes are updated, the decoding process may have completed on decoding iteration. The concurrent trigger of the computational nodes 704 and 706 may be repeated to achieve one or more iterations of the decoding process.

Please replace paragraph [00053] on page 24 with the following paragraph:

[00053] The steps of a method or algorithm described in connection with the embodiments disclosed herein may be embodied directly in hardware, in a software module executed by a processor, or in a combination. A software module may reside in ~~RAM memory~~ Random Access Memory (RAM), flash memory, ~~ROM memory~~ Read Only Memory (ROM), ~~EPROM memory~~ Erasable Programmable Read Only Memory (EPROM), ~~EEPROM memory~~ Electrically Erasable Programmable Read Only Memory (EEPROM), registers, hard disk, a removable disk, a [[CD-ROM]] Compact Disk Read Only Memory (EEPROM), or any other form of storage medium known in the art. An exemplary storage medium is coupled to the processor such that the processor can read information from, and write information to, the storage medium. In the alternative, the storage medium may be integral to the processor. The processor and the storage medium may reside in an ASIC. The ASIC may reside in a user terminal. In the alternative, the processor and the storage medium may reside as discrete components in a user terminal.